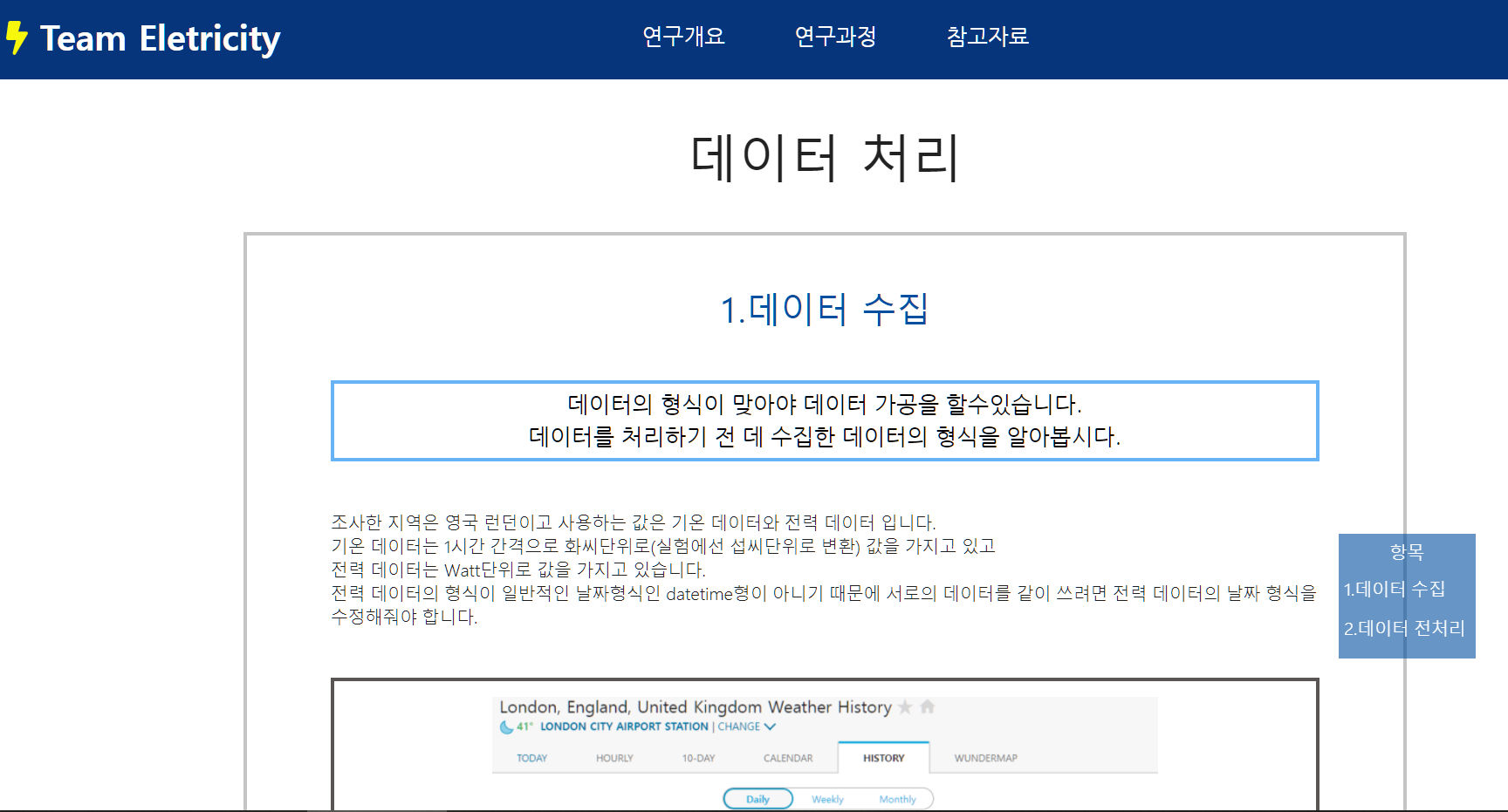
2월18일 전력 예측 페이지 해야할 일

1. 딥러닝 모델,그래프 예측 페이지,참고 자료 페이지 설계도 제작
2. 노드js 강의 듣고 변형된 구글 차트 사용하기
3. 설계도 바탕으로 연구 과정 페이지 제작하기

2월19일~20일 한일: 데이터 처리 페이지 제작함

1. 그림 뒤에 유니코드로(&#9650;) 화살표 표현
2. 그림 사이즈는 width:100%로 하면 화면에 안 빠져나옴, float:left or right 로 정렬 위치 배치 가능함



2월21일 딥러닝 모델 페이지 제작 중:

목차마다 선택 박스를 만들고 버튼을 누르면 해당하는 목차의 내용이 나오게 구현할 예정  
내가 구현하고 싶은 건 마우스 한 번 누르면 열리면 한 번더 누르면 닫히는식으로 구현하는 것임 그리고 하단에 닫기 창으로 스크롤 다 내려도 닫을 수 있게 구현함

텍스트이(가) 표시된 사진

자동 생성된 설명

----마우스에 반응 하는 창을 만들기 위해서 js 공부 병행함----

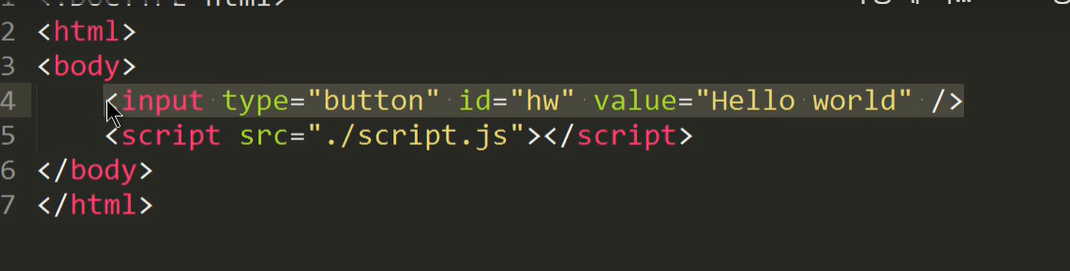
텍스트이(가) 표시된 사진

자동 생성된 설명

Id가 ‘hw’인 값을 못 찾아서 null값이 담기게됨

Window.onload = **function**(){} 웹내용을 로드한 후에 요소를 찾게 됨

body뒤 에 자바 스크립트를 입력하면 렌더링을 1 번더 걸치지 않기 때문에 빠름



마우스에 반응 하는 코드들

<ul>

        <li class="menu">

            <a><img src="" alt="상위메뉴이미지2"/></a>

            <ul class="hide">

                <li>메뉴2-1</li>

                <li>메뉴2-2</li>

                <li>메뉴2-3</li>

                <li>메뉴2-4</li>

                <li>메뉴2-5</li>

                <li>메뉴2-6</li>

            </ul>

        </li>

    </ul>

    <script>

        $(document).ready(function(){

            //id가 menu인 내용안의 a 태그를 클릭했을 때

            $(".menu>a").click(function(){

                //다음 ul태그의 hide 클래스를 빼고 한 번 더 누르면 넣는다.

                // $(this).next("ul").toggleClass("hide");

                var sub\_menu =$(*this*).next("ul");

                //sub\_menu가 화면상에 보일 때는 위로 부드럽게 펼치기

                if(sub\_menu.is(":visible")){

                    sub\_menu.slideUp();

                }else{

                    sub\_menu.slideDown();

                }

            }).mouseover(function(){

                //마우스 오버되면 메뉴 자동으로 펼치기

                $(*this*).next("ul").slideDown();

            });

        });

    </script>

2월22일 할 일:마우스에 반응하는 코드들로 네비게이션이랑 딥러닝 모델 페이지 제작

텍스트이(가) 표시된 사진

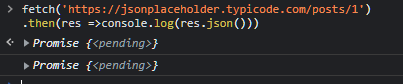
자동 생성된 설명

자바 스크립트로 구현 완료

2월23일 할 일: 그래프 예측 페이지 설계 및 관련 공부 병행

딥러닝 페이지는 완성함

Fetch()함수로 원격 api 호출하기 (get방식)



Body가 스트림개체여서 json 개체로 호출해주는 것이 용이함

Post방식 api 호출  
  
텍스트이(가) 표시된 사진

자동 생성된 설명

get방식과 동일하게 json 형식 사용함  
.then(data=> console.log(data)) 데이터를 console에 출력함

텍스트, 화면, 스크린샷, 은색이(가) 표시된 사진

자동 생성된 설명

전달해준 foo, bar 게시물에 id가 101로 부여됨

Let은 재선언 불가능하지만 재할당은 가능

const name = 'bathingape'

console.log(name) // bathingape

const name = 'javascript'

console.log(name)

// Uncaught SyntaxError: Identifier 'name' has already been declared

name = 'react'

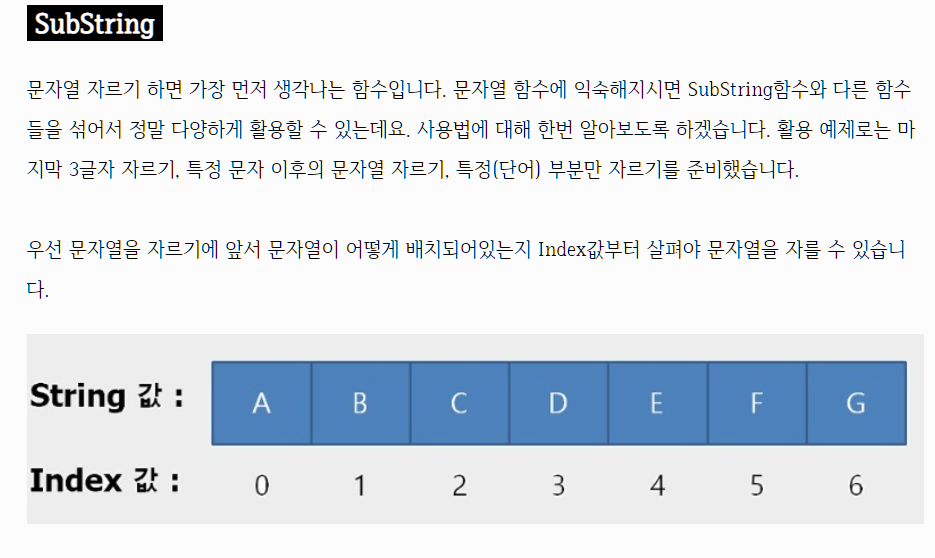
console.log(name)

//Uncaught TypeError: Assignment to constant variable.

cmd창에서 netstat -a -n -o 현재 사용중인 포트 봄

taskkill/f /pid 000000 해당 포트 중지

substring 문자열을 자르는 함수



노드 js 종료 ctrl+c

2월28일 할 일:

1. x축 값이 제대로 나오게 하기 0
2. 제목에 ‘x월’ 표시 나오게 하기 0
3. 그래프 색상 변경하기 0

1.Csv 파일 끝에 비어 있으면 에러 남 여백 없애서 처리 완료

텍스트이(가) 표시된 사진

자동 생성된 설명

모델 2개 그래프 웹에 담기 성공

텍스트, 필기구, 문구, 연필이(가) 표시된 사진

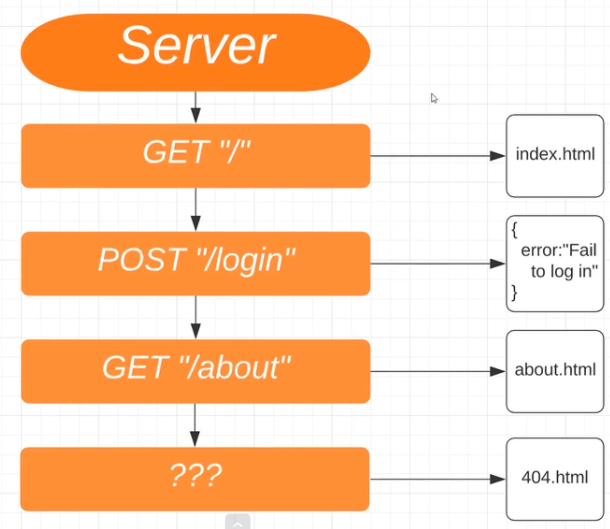
자동 생성된 설명

3.참고 자료 페이지 제작함

텍스트이(가) 표시된 사진

자동 생성된 설명

노드 js 연습 예제



const express =require('express')

const server=express()

const port=3000

server.get("/",(req,res)=>{

    res.sendFile(\_\_dirname+"/index.html");

});

//get 메소드가 들어오면 inde.html을 보낼 수 있게 됨

server.get("/about",(req,res)=>{

    res.sendFile(\_\_dirname+"about.html");

})

server.listen(port,(err)=>{

    if(err) return console.log(err);

    console.log("the server is listening on port 3000");

});

Ctrl+shift+p vscode에서 깃클론 열기

이미지 확대 코드

.img-wrap img {

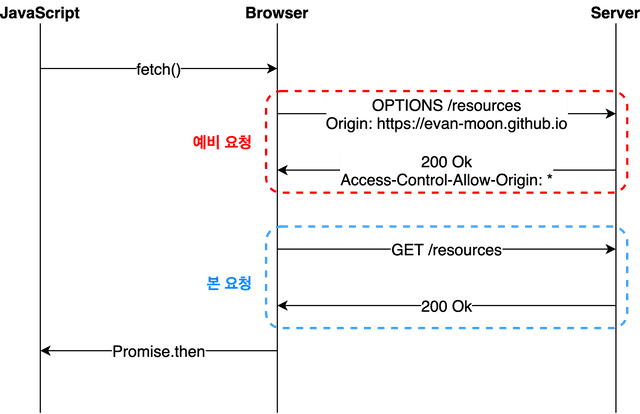
1. transform: scale(1.1);
2.  transition: transform .8s ease-in-out;

}

}

3월 1일 한 일

Cors 프리 플라이트(예비 요청)전송



우리가 자바스크립트의 **fetch** API를 사용하여 브라우저에게 리소스를 받아오라는 명령을 내리면 브라우저는 서버에게 예비 요청을 먼저 보내고, 서버는 이 예비 요청에 대한 응답으로 현재 자신이 어떤 것들을 허용하고, 어떤 것들을 금지하고 있는지에 대한 정보를 응답 헤더에 담아서 브라우저에게 다시 보내주게 된다.

이후 브라우저는 자신이 보낸 예비 요청과 서버가 응답에 담아준 허용 정책을 비교한 후, 이 요청을 보내는 것이 안전하다고 판단되면 같은 엔드포인트로 다시 본 요청을 보내게 된다. 이후 서버가 이 본 요청에 대한 응답을 하면 브라우저는 최종적으로 이 응답 데이터를 자바스크립트에게 넘겨준다.